



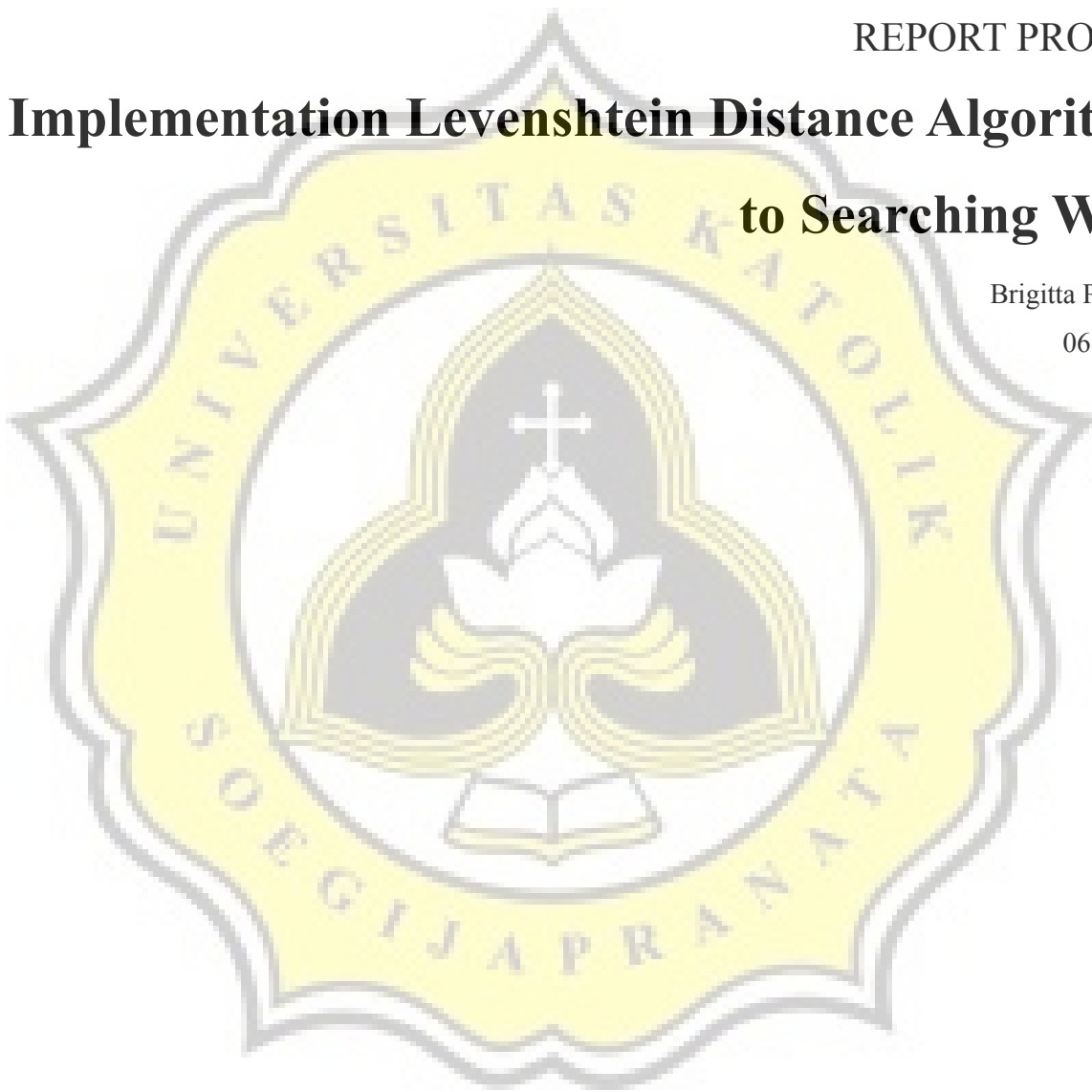
REPORT PROJECT

# **Implementation Levenshtein Distance Algorithm to Searching Word**

Brigitta Petrastuti

06.02.0020

2011



**FAKULTAS ILMU KOMPUTER  
UNIVERSITAS KATOLIK SOEGIJAPRANATA**

Jl. Pawiyatan Luhur IV/1, Bendan Duwur, SEMARANG 50234

Telp. 204-8441555 (hunting) Web: <http://www.unika.ac.id/>

Email: [ikom@unika.ac.id](mailto:ikom@unika.ac.id)

# APPROVAL AND RATIFICATION PAGE

## PROJECT REPORT Implementation Levenshtein Distance Algorithm to Searching Word

This Project Report has been approved and ratified by Dean of Computer Science Faculty on

.....

With the approval,

Examiner,

Examiner,

Rosita Herawati ST, M.IT  
NIP: 058.1.2004.263

Robertus Aji Setiawan, ST, McompIT  
NIP: 058.1.2004.264

Examiner,

Hironimus Marlon Leong, S.Kom, M.Kom  
NIP: 058.1..2007..273

Supervisor,

Dean of  
Faculty Computer Science

Gregorius Hendita AK,  
NIP: 058.1.2008.277

Hironimus Marlon Leong, S.Kom, M.Kom  
NIP: 058.1..2007..273

## STATEMENT of ORIGINALY

The undersigned:

Name : Brigitta Petrastuti

ID : 06.02.0020

Here by certify that this project was made by my self and not copy or plagiarizes from other people, except that in writing expressed to the other article.

If it proven that this project was plagiarizes or copy the other, I'm ready to accept a saction.

Semarang, January 15<sup>th</sup>, 2011

Brigitta Petrastuti

06.02.0020

## FOREWORD

I dedicated this graduation to my whole family that had been support me.

I would like to say thank you to:

- to God, that always guide me at making this project,
- to my parents, who always pray for me and support me to finish my project,
- to Gregorius Hendita A.K, as my supervisor who always helping, guiding and give me the ideas to finish this project,
- to my friends that always help me to support and helping me.

Last, I want to apologize if I've made some mistake at finishing this project and writing this respon. Therefore, critics and suggestions are expected.

Semarang, January 15<sup>th</sup>, 2011

Brigitta Petrastuti

## ABSTRACTION

Levenshtein distance is a measure of the similarity between two strings, which we will refer to as the source string (s) and the target string (t). The distance is the number of deletions, insertions, or substitutions required to transform s into t. The greater the Levenshtein distance, the more different the strings are.

Levenshtein distance is named after the Russian scientist, Vladimir Levenshtein, who devised the algorithm in 1965. If you can't spell or pronounce Levenshtein, the metric is also sometimes called edit distance. The Levenshtein distance algorithm has been used in spell checking, speech recognition, DNA analysis, plagiarism detection .

This project is used for searching word from one file and after that this program will counting the word that was found, so users in addition to knowing whether the word exists or not, user will know too how much word in file whose user inputed. User only push the open button or user can choose the menu and click open in menu file, and choose the file that user want to search. After then, user can fill the keyword in text field, and then push the search button. So, the program will be printed result, the word is exist or not exist in the file, and if exist the result can be shown too how many word that exist in that file.

This project, is made using java language and levenshtein distance algorithm. However, this project in rough and still own feebleness which need corrected.

For example, if string (s) is "test" and string (t) is "test", then levenshtein distance of string(s,t) is 0, because no transformations are needed. The strings are already identical.

Another example if string (s) is "test" and string (t) is "tent", then

levenshtein distance if string(s,t) is 1, because one substitution (change "s" to "n") is sufficient to transform string (s) into string (t).

Keyword: searching word, levenshtein distance algorithm



## TABLE OF CONTENT

APPROVAL AND RATIFICATION PAGE.....	ii
STATEMENT of ORIGINALY.....	iii
FOREWORD.....	iv
ABSTRACTION.....	v
TABLE OF CONTENT.....	vi
TABLE OF FIGURES.....	vii
TABLE OF TABLES.....	viii
CHAPTER I INTRODUCTION	
1.1 Introduction.....	1
1.2. Scope.....	1
1.3. Objective.....	1
CHAPTER II LITERATURE STUDY	
2.1. Algorithm.....	3
CHAPTER III PLANNING	
3.1. Research methodologies.....	8
3.2. Project Management.....	8
CHAPTER IV ANALYSIS AND DESIGN	
4.1. Analysis.....	9
4.1.1. Use Case.....	9
4.2. Design.....	10
4.2.1. Class Diagram.....	10
4.2.2. Data Flow Diagram.....	13
CHAPTER V IMPLEMENTATION AND TESTING	
5.1. Implementation.....	15
5.2. Testing.....	20

## CHAPTER VI CONCLUSION AND FURTHER RESEARCH

6.1. Conclusion.....	25
6.2. Further Research.....	25
CHAPTER VII REFERENCES.....	26





## TABLE OF FIGURES

Figure 4.1 Usecase Diagram.....	9
Figure 4.2 Class Diagram Structure.....	10
Figure 4.3 Data Flow Diagram level 0.....	13
Figure 4.4 Data Flow Diagram level 1.....	13
Figure 4.5 Data Flow Diagram level 1.....	14
Figure 5.1 Open File Chooser.....	15
Figure 5.2 Method to Open File.....	16
Figure 5.3 Read Sentences by word.....	16
Figure 5.4 Set Length of Matrix.....	17
Figure 5.5 Calculation .....	17
Figure 5.6 Count the Keyword that was Found .....	18
Figure 5.7 Progress Bar .....	18
Figure 5.8 View The Content of File.....	19
Figure 5.9 View The Content of File.....	19
Figure 5.10 Start The Program.....	20
Figure 5.11 Select File.....	21
Figure 5.12 File that was Selected.....	22
Figure 5.13 Input Keyword.....	22
Figure 5.14 Output in Terminal.....	23
Figure 5.6 Output.....	24

## TABLE OF TABLES

Table 2.1 steps and description of levenshtein distance.....	4
Table 2.2 Step 1 and 2 .....	4
Table 2.3 Steps 3 to 6 When $i = 1$ .....	5
Table 2.4 Steps 3 to 6 When $i = 2$ .....	5
Table 2.5 Steps 3 to 6 When $i = 3$ .....	5
Table 2.6 Steps 3 to 6 When $i = 4$ .....	6
Table 2.7 Steps 3 to 6 When $i = 5$ .....	6
Table 2.8 Steps 3 to 6 When $i = 6$ .....	6
Table 2.9 Steps 7.....	7
Table 3.1 Table of Time.....	8
Table 4.1 Class Diagram KeySearch.....	11
Table 4.2 Class Diagram Searching.....	12